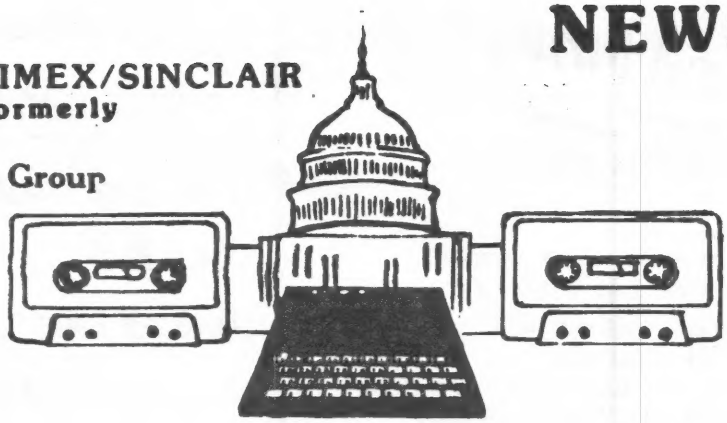
CATS NEWSLETTER

CAPITOL AREA TIMEX/SINCLAIR USERS GROUP :Formerly
Prince George's Timex/Sinclair User's Group



April 1985
Vol. 3, No. 1

* * * CONTENTS * * *

PRESIDENT'S COLUMN

Writing For the 'Letter

Once again, articles are coming in from a variety of sources. Not all of them made it into this issue, but I'm standing behind my policy of printing everything that's submitted by members. If you don't see it in this issue, look for it next month. Another newsletter said; "If you don't see your article, you probably didn't write one!" Once again, anything you've got is fine - especially REVIEWS, or short to medium length programs.

Future Business

The CATS library is here! We have a tape of public domain 2068 programs that will be available at the next meeting. The programs run from simple graphics demos to Ned Beeler's mailing list program. If you want a copy, bring your recorder, patch cord, and a C60 or 90 tape to the next meeting, and some cooperative member will oversee the use of the tape to tape duplicating equipment. Right now, we can make three dups at once, but that probably can be expanded.

If you have programs to donate, bring them to the next meeting also, and we'll compile them for the one after that. Public Domain means homebrew programs, and those that have been typed in from magazines.

The "Executive" Committee

I've been hosting a meeting of interested club members, every Monday evening at 7:30. We generally end up with a mix of club business and individual problems or triumphs.

Last week, Tom Bent came and demonstrated his method of modifying the 1000 for faster LOADing. His method is somewhat original - he uses nippers to lop off the upper half of the filter cap (C11) on the mic output. He also had a number of other ways to prune your 2068 for better performance; for stronger SAVEs on the 2068, clip C 10; for more sensitivity on LOAD, change R 12 from 6.8K to 1.8K.

This week, Murray Barasch brought a program he was trying to debug, and we got it going, as well as polishing up the newsletter for the printer.

Everyone is invited to these meetings - if you've got time, come on down!

Errata

The IRSCALC program last month was a great way to get that ol' 1040 done, except that several of the formulas I listed were wrong. The correct ones are:

```
F6  B07+C07+D07+E07
F9  I07+J07+K07+L07+M07
F11 T07+U07+V07+W07
F20 H11+I11
```

Last Meeting.... and Next.

The March meeting went very well. Tom Bent showed an adapter to use joysticks with Spectrum programs, and Mike Cohen demo'd his baby MacIntosh. Next month... is up in the air, as usual. I'm hoping to get a QL, but no promises. If you've got something to show, to the whole group or just a portion, let me know and we'll make room. We may hold an advanced seminar on the Timex hardware on the day following - details to be announced.

See you there!

Mark Fisher

Attention!

SPECIAL INTEREST ACTIVITIES

At the last meeting, seven to eight people indicated interest in each of two activities: MACHINE CODE (or assembly language) and BUSINESS and INVESTMENT applications. John Conger will again lead the applications group and is devising a topics schedule. Fisher, Conger, Bent and others are devising a curriculum and selecting text materials for the machine language course.

All those who signed up will be contacted before the programs jell to assure suitable direction and level to meet member's interests. Others interested may sign up at the April meeting and we expect to get going on a regular basis in mid April. Anyone with suggestions for either group can call Conger or Fisher in the evenings. Conger: 654 5751; Fisher: 589 7407

SUBMISSIONS for this newsletter are eagerly solicited. First priority will be given to member's submissions. Publication of material does not transfer rights from the author, in fact, it may establish priority.

Submissions may be reviews, articles on applications, programming techniques, hardware, or anything else you can imagine. Pertinent articles from other publications will also be considered.

Bring material to the meeting, or send it to PO box 725, Bladensburg, MD 20710. I would prefer material to be typed, single spaced, in 3½" columns - but don't break your back: the Xerox dosen't really care. Printouts from the 2040 printer are fine, but, use Radio Shack paper, and don't put scotch tape over the printing.

IMPORTANT DATES:

Newsletter d/l	Meeting
	April 13
April 19	May 11
May 17	June 8
June 17	July 13

See you there!

CATS ROSTER OF OFFICERS

President	Mark Fisher
Vice President	Mike Cohen
2nd Vice Pres.	Wayson Lee
Secretary	Bob Curnutt
Treasurer	Sarah Fisher

Newsletter Team

Production Editor:
Sarah Fisher
Manuscript Editor
Mark Fisher
Post Office liason:
Jules Gesang
Writers:
Anne Andersen
Anon.
Tony Brooks
John Conger
Lamont Downs
Mark Fisher
Sarah Fisher
Bob Lussier
Mike Morris
Roald Schrack
Vernon Smith
Stew Vance
Bill Ware

RESOURCES

Just one publication this month, a tried and true journal, and still going strong.

QZX (HAM radio & Sinclair computer) $12:00
Alex Burr
2025 O'Donnell Dr.
Las Cruces, NM 88001

MY WORD!

By John Conger

THE COMPUTER ENTREPRENEURS

By Levering, Katz & Moskowitz

In its start-up years, the automobile industry had hundreds of back yard tinkerers bolting metal together to build a motorized vehicle that would make their fortunes. We know of Henry Ford and Walter Chrysler, but who remembers the contributions (let alone the names) of the many others? This book makes the laudable attempt to profile the Personal Compuer industry at its start, before the shake out that is expected to reduce the hundreds now active to the dozen likely to survive.

Sixty five entrepreneurs are sketched in brief biographies and some data that tell us where they came from and how they failed or made millions - or both. Not just computer designers and software wizards are covered. The contents give almost equal space to peripheral providers like Hayes modems and Tandon disk drives; Shopkeepers like Millard, the first compter billionaire and owner of ComputerLand; information moguls like Brunnell of PC World and Wilkins of CompuServe; moneybags and gurus like Rock, McKenna and venture capitalist Perkins. One Japanese is covered although a separate book would be required for the hundreds active there.

As a reference source of "who done it", it is useful for those of us who like to follow industry developments. However, it is marred by two flaws, the main one being who is left out and the second being the lack of dates when many signifiant events occurred.

Those who have read Fire in the Valley will have a more complete picture of the flow of history than can be presented by the 65 biographies in the subject book. The Computer Entrepreneurs does not include (or gives only passing mention to) Ed Roberts who really started it all with his Altair 8080 kits. And the information moguls section omits Les Solomon, technical editor of Popular Electronics (now Computers and Electronics). These two men gave the personal computer its start. Another missing name is Robert Noyce of Intel who produced the 8080 chip and some computers using it, before the Altair. The compters were for his stable of programmers, like Gary Kildall, to use in their work. The names of these men are not highlighted as I feel they should be. And of course Sinclair is not mentioned at all.

Still, the stories that are told are facinating: Tramiel, a Polish survivor of Auschwitz who built a typewriter repair shop into Commdore; Mitch Kapor the transcendental meditation teacher who saw his Lotus 123 program humble Visicalc; Phillip Hwang the Korean immigrant dishwsher now worth half a billion; and of course the career highlihts of well known personalities such as Steve Jobs, George Morrow, Andrew Kay, Adam Osborne; Bill Gates and George Tate.

This book turns out to be more of a "people" book than the industry profile it set out to be. As such it has much interesting human interest content, although it also has many holes in it due to the chosen format.

All in all, if you have read Fire in the Valley then you have read much of the material covered here, and "Fire" is the more complete and competent history of the industry as well as providing more depth and color to the characters involved.

MORE SPECTRUM FEEDBACK, #1

Listed below are a number of Spectrum Programs which I have run on the 2068 with an Emulator. They should run on a 2068 with Romswitch, but I make no promises!.

Match day.- Soccer simulation with superb graphics
Spectrum cricket.
Arcade sampler.-Three simple arcade games.
War of the Worlds.
Sabre Wulf.-Voted best game of 1984 in U.K.
The fall of Rome.-Defend the Roman Empire for 50 years.
Football manager.-Coach a soccer team.
King Arthers quest.-Yet another adventure game.
Battlecars.
Apocalypse.
Select 1.-12 games for 12 Pounds (Hunchback,Spectres,Time Gate,Space Invaders,Mr Wimpy,Moon Buggy,Denis,Missile Defense,Pool,Tranverse,Meteor Storm)
Deus Ex Machina.-Comes with matching music tape.
Alchemist.
Mini office.-This tape includes a word Processor,spreadsheet, database, and graphics Plotter. The word Processor will not run using an Emulator.

I've seen a great deal of nonsense in Print about the Emulator with claims that it will run only 60% of Spectrum software. This is just not true, my estimate is at least 95%.

I buy all my Spectrum software direct from England. Most British software suppliers will send tapes direct to the U.S.A. for cost Plus 1 or 2 Pounds extra Postage. Just look in magazines such as ZX Computing and Sinclair User. All you need to do is quote your credit card number. The cost advantage is obvious, most Spectrum software costs 3 to 10 Pounds compared with $20 and up in the U.S.A.

Regards
Tony Brooks

P.S. This letter was done using Mscript adapted to drive the Tasman interface. The Printer used was a Gorilla/Banana.

MORE SPECTRUM FEEDBACK, #2.

Here is a list of my programs that RUN on the T/S 2068 and the G. Russell ROMSWITCH.

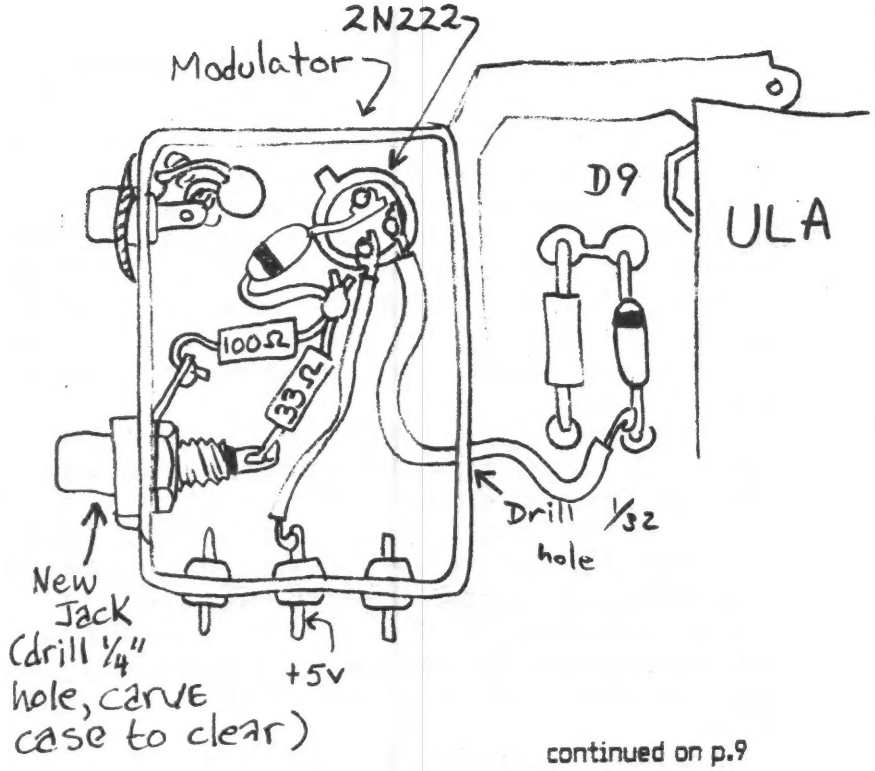
(01) Eye of the Star Warrior
(02) Darts
(03) Poker-Bandit
(04) ZX Inlay Card
(05) Zodiac
(06) Daley Thomson's Decathalon
(07) 5 Print Styles
(08) MUGSY
(09) Jet Set Willy
(10) Cribbage
(11) Typesetter
(12) 3D Space Art
(13) Stargazer's Secret
(14) 3 Tape COOKBOOKS
(15) Ad Astra
(16) Speedyload
(17) T.V. Scrabble
(18) Micro Olympics
(19) Pinball Wizard
(20) Zaxxan
(21) Booty
(22) Scuba Dive
(23) Matchpoint
(24) Tornado Low Level
(25) Cyclone
(26) Fighter Pilot
(27) 3D Starstrike
(28) Lords of Midnight
(29) Nightflite II
(30) Codename Mat
(31) 3D Tunnel
(32) Snake Pit
(33) Colour Clash
(34) Shark Attack
(35) Galactic Trooper
(36) Tarot (Text)
(37) Cosmos
(38) Planetarium
(39) The Solar System
(40) The Night Sky
(41) Geography I & II
(42) Guitar Tutor I
(43) Astronomer
(44) Delta Wing

continued on p.5

DRIVING A MONITOR
From the T/S 1000

The T/S 1000 was designed to produce useable results on a standard TV. Its display was limited to 32 character lines, partly to ensure that the letters were still legible. They are larger, but if you spend a lot of time in front of the set, the blurriness of the standard TV screen can be wearing. The fault lies in the circuits of the reciever. A standard TV is designed to create a pleasing image from 6 feet - while at 18 inches, the features are often blurred.

A monitor, on the other hand, has been designed to produce a crisp image up close. Unfortunately, the average monitor requires a different signal than a standard TV, and the 1000 does not offer a suitable output. Like many other things with this machine, this can be changed! The correct signal is present, but is too weak to drive a monitor. The mod involves making three solder connections within the machine, to add a one transistor amplifer for the signal.



continued on p.9

The Mod...

Materials required are:

2N222 Transistor (Radio Shack sells these, but theirs are of variable quality. A Motorola version will have a higher gain and a crisper output than the average Radio Shack specimen.)
33 Ω, 1/4 watt resistor
100 Ω, 1/4 watt resistor
IN914 glass diode. Don't worry about the precise part number - all those little fellers are about the same.
A panel mount RCA Jack (ex. Radio Shack 274-346)

The circuit can be put almost anywhere. There is room to fit it inside the modulator without affecting its operation, thus leaving an uncluttered machine, & the option of either TV or monitor output. It involves drilling holes very near to some resistors in the modulator, but I think that the results are worth it.

To install the circuit, first remove the ULA and Z80 chips, and store them in aluminum foil (to protect against static electricity). Next, carefully drill the 1/4" hole for the jack, and the 1/16 hole for the signal line. Solder up the assembly outside the case, and install it, being careful not to leave any solder where it shouldn't be. Carve the case to fit the new jack, insert the chips, and enjoy!

continued from p.4

(45) Eddie Kidd Jump Challenge
(46) MAH JONG
(47) River Raid
(48) Black Crystal
(49) Address Manager
(50) TUBECUBE (3D Rubik's Cube)
(51) The HOBBIT
(52) Psytron
(53) Masterfile
(54) Electronic Designer
(55) Machine Code Tutor
(56) Skool Daze
(57) 007 Spy Tape Copier
(58) Lerm Tape Copier 6
(59) Omnicalc II
(60) Super Re-Number
(61) Supercode II
(62) Spectrographics
(63) Superchess 3.0
(64) Zombie Zombie (sequel to 3D Ant Attack)

A few Spectrum programs that would not RUN on the ROM-SWITCH.

(1) 3D Dracman
(2) Legend of Avalon
(3) Doomdark's Revenge (Sequel to Lords of Midnight)
(4) Eights
(5) Vagan Attack

R. Lussier
MEMBER

For 2068 Users Only: DRESS UP YOUR GAME PROGRAMS WITH A SPLASH OF COLOR

by Lamont Downs

Here's a short program that you can merge with almost any game-type program and which enables you to change the paper color of the entire screen in a flash WITHOUT AFFECTING ANYTHING ELSE ON THE SCREEN. For example, if you've got a spaceship traveling across a black sky you can cause the entire sky to flash red for an instant and then back to black without erasing anything or changing any of the ink colors - and do it without having to remember any complicated USRs or even line numbers!

To use, first type in the program below and SAVE it. Then LOAD (or type in) the program you want to "decorate." Be sure the program does not have a line 1 or any lines from 9970 to 9989 (if it does, relocate them). Then MERGE the program below.

Now all you have to do to change paper color anywhere in the program is to insert the command GOSUB followed by the first three letters of the color (GO SUB blu, GO SUB gre, etc.). For example, if you have a blue sky background, you can light up the sky with a red flash with the following program line:

GO SUB red:PAUSE 2:GO SUB blu

You can create dandy explosions by using sequences of colors (red-green-yellow-green-red) etc.). Some moniters will "tear" if colors change too quickly, so you may need to experiment with the length of the pauses for best effect. And don't forget you can throw in sound effects using the SOUND command instead of the PAUSE.

One last warning: until line 1 is executed the machine code for changing color is not loaded into memory and the program will crash if you try to change colors, so if you're starting the program somewhere other than line 1 be sure that you take this into account.

"Splash"

```
  1 CLEAR 65279: GO SUB 9970
9970 RESTORE
9972 FOR n=65280 TO 65304: READ
x: POKE n,x: NEXT n
9974 LET bla=9980: LET blu=9981:
 LET red=9982: LET mag=9983: LET
 gre=9984: LET cya=9985: LET yel
=9986: LET whi=9987
9976 RETURN
9978 DATA 1,192,2,33,0,88,126,20
3,0,203,0,203,0,119,35,62,0,11,1
85,32,241,184,32,238,201
9980 POKE 65288,159: POKE 65290,
167: POKE 65292,175: LET rand=US
R 65280: RETURN
9981 POKE 65288,223: POKE 65290,
167: POKE 65292,175: LET rand=US
R 65280: RETURN
9982 POKE 65288,159: POKE 65290,
231: POKE 65292,175: LET rand=US
R 65280: RETURN
9983 POKE 65288,223: POKE 65290,
231: POKE 65292,175: LET rand=US
R 65280: RETURN
9984 POKE 65288,159: POKE 65290,
167: POKE 65292,239: LET rand=US
R 65280: RETURN
9985 POKE 65288,223: POKE 65290,
167: POKE 65292,239: LET rand=US
R 65280: RETURN
9986 POKE 65288,159: POKE 65290,
231: POKE 65292,239: LET rand=US
R 65280: RETURN
9987 POKE 65288,223: POKE 65290,
231: POKE 65292,239: LET rand=US
R 65280: RETURN
```

TASWORD TWO

When I was asked to write about the Tasword Two word processing program I'd just received from England, my first thought was that I lacked the proper background to evaluate the program's merits. I was the guy that spent a week learning to operate the company word processor but still writes out manuscripts. Give it to a typist. So I faced Tasword with a certain amount of trepidation. After using it a while, I have to give it a top grade in my book. In writing this article I received some unexpected help from Tom Ferrebee's article in Vol. 11 of TS Horizons. I hope Tom won't mind but I have reproduced his excellent outline of the system's features.

The system loads in just under 2 minutes and starts with a blinking cursor in the upper left corner of the screen, which is 64 characters wide. An "information" line at the bottom gives the line number and column position of the cursor, tells if right justification, word wrap, and the insert mode is ON or Off, and, lastly, to push EDIT if help is needed. For first time users there is an excellent tutorial that quickly teaches how to use the numerous functions of the program. The help screen contains a listing of the "normal" mode commands, all of which are accessed by holding down one of the shift keys and the applicable function key. The second, or "extended" mode, is accessed by pushing the symbol and caps shift keys simultaneously. These commands are mainly used for formatting and some miscellaneous functions. The execution of these commands is also different from the normal mode, since once you are in the extended mode, you only have to press a single key, but the system reverts to the normal mode after the key stroke.

Tasword differs from MSCRIPT in that it supports a ZX printer, as well as a full sized printer utilizing a Centronics parallel interface. When you are ready to print using a full sized printer, you access the Print screen by using the Stop command. You then have the option of defining your graphics or using the Epson FX80 character codes which are imbedded in the program. This screen is also used to load or save text files.

TASWORD TWO SPECIFICATIONS

Screen size	64x22
Text file size	19,200
Printers	Both
Interface	Centronics
Cursor by	Letter, word, line, or screen
Delete by	Letter, word, or line
Margins	Left or right
Block	Move or copy
Search	Find or replace
Wordwrap	Yes
Right justify	Yes
Centering	Yes
Line length	64
Spacing	1,2,3+
Tutorial	Yes

My only wish is that Tasword Two had a Tab capability. It is a little awkward to have to pound the space bar to get paragraph indentations. It is, however, an excellent WP program. Now that I've gotten used to it, it's fun to use. Goodby, paper and pen!

Vernon Smith

Inside the machine

For the average computer owner, canned programs or home-brew programs in BASIC will do the jobs we had in mind when we bought the computer. At one time or another, however, all of us will have a need to know something of the inner workings of our purchase. This may be the result of a need to do things that BASIC can't do - such as control I/O ports to an alarm system for example - or it may be a need to remove some of the sense of uncontrollable magic from the machine. While replaceing this feeling of mystery, we will gain an appreciation of at least four interlocking levels of complexity in the machine. Starting from the smallest, they are:

1. The actual hardware: wires and semi-conductor switches.
2. The machine code that controls these switches.
3. The BASIC (or other high level language) that lets you easily invoke the machine code, and...
4. The real world problems that are addressed by your programs.

THE REAL WORLD

Now let's work backwards through those levels. The real world offers an infinite number of situations that need some sort of calculation. If you need assistance, you go to an expert, who asks you questions, then provides an answer. Some canned programs operate at this level - they ask you questions, you respond, and they provide you with the results.

HIGH LEVEL LANGUAGES

Over the last 10,000 years, we have developed symbolic tools to assist us. These include the alphabet and language, and the number system and algebra. The real world tools can be adapted to the computer. The result is a programming language. It uses (in the case of BASIC) English words to describe its functions, and algebraic notation to express the mathematical operations. [Neither the English, nor the algebraic notation conform to the original, "real world" standard. They have been modified to accomodate the remaining two levels of complexity.] At this level, numbers are represented as you would expect - "10" means ten, while some of the symbols are new - such as * for times. Generally, this is a self-sufficient level - you needn't look any farther when using or programming the machine. Some programs can be so complex as to approach a language themselves in flexibility and power - DBII, or even VU-CALC are examples.

MACHINE CODE

The third level, machine code, is the level that actually controls the central processor. In the Timex there is an extremely complex program of 8192 steps that is responsible for the operation of the computer, as well as the interpretation of any BASIC program that might be present. These are two very different jobs, and in other computers they are performed by two different programs.

The first job is that of an operating system. This portion coordinates the various elements of hardware, allowing for display, printouts, loading, and saving of data. The second portion is the BASIC interpreter and editor. This is the "brain" that you are fighting as you try to avoid the dreaded inverse "S". This interpreter can be replaced: for example, the egregious BASIC that comes with the Commodore can be replaced with a much more efficient version (though it's still not as compact as Sinclair BASIC).

At this level, numbers are represented in whatever way is most convenient for the program. In its reports to the display file numbers are expressed as decimal numbers, but for its own use, there are four different methods of expression - each used when most appropriate. It is possible to write your own programs in machine code. It requires extensive knowledge of these deeper levels of the computer, but also offers enormous freedom and speed. You can make the computer execute machine code by using "USR nnn", where nnn is the starting address of the code.

HARDWARE

Numbers as Voltage

In this deepest level all information is in the form of electrical potential. A +5 volt signal might represent a "one", while a ground level might represent a "zero". This might seem very limiting; it is. Single lines carry information around the computer, controlling the timing and operation of its "hidden" accessories. But there isn't any way to have an individual line for each possible number. Rather, there are two groups of lines, known as busses, that carry numbers. These busses are organized as sets of eight lines. The first bus is the data bus; it consists one set of eight. [The second bus is the address bus - we will get to it in a moment.] There are 256 different possible patterns of voltage in these eight lines, ranging from "00000000", "00000001", "00000010", "00000011",... to "11111111". This group of eight is called a byte. To express a number larger than 255, you need a second byte. With two bytes, you have 256*256 possible numbers, or 65536. This is where the magic 64K number comes from. (each K=2*2*2*2*2*2*2*2, or 1024)

The "Real" Computer

Now that we have a way to express numbers electrically, we need to have some device that will perform operations with these numbers, such as addition. This is the job of the Central Processing Unit. Originally, this was a device the size of a refrigerator. In a micro, it's on a chip, called a microprocessor. It is able to take a series of numbers from the memory (the machine code) and use them as instructions, controlling the actions of the rest of the computer, as well as performing simple arithmetic and logical steps.

The CPU has a very few locations to hold numbers, ready to be used. These locations are called registers. Each register, and the controlling logic that goes with it, are composed of networks of transistors. In the Z80, data is handled in one byte units. Other chips use two bytes (16 bits) or four bytes (32 bits) for data.

For each byte of machine code that must be executed, the CPU must perform a number of discrete steps. These steps are held as microcode, which is yet a deeper layer of operation. Since it is common to all Z80 CPU chips and cannot be changed, we can ignore it.

Storage

A computer must store many more numbers than could possibly be held in registers, however. To store these numbers, the computer has additional chips. These form a series of banks of eight switches. Each bank can hold one number, between 0 and 255. Each bank has a unique address. (You can look at each bank from BASIC. Try the first bank, with address zero. "PRINT PEEK 0" should give 211.) These banks are called up through the second bus, the address bus. Since our computer needs more than 256 spaces to store numbers, the address bus is two bytes wide, allowing 64K different addresses. (Not all these addresses must be used.)

Methods of Storage

ROMs

There are three ways of physically building these banks. They can be jumpers, capacitors, or transistor networks. Banks of jumpers are called ROMs. Numbers are stored as a pattern of connections - intact lines will allow the voltage to build up to +5V, while open circuits leave the line at 0V. These patterns can only be created in in a factory - as far as you are concerned, they are Read Only Memory. This is ideal for a program that always must accompany the machine, and that is why the Timex comes with its operating system in ROM. These can never be erased.

EPROMs

Capacitors can be loaded with numbers at will, by applying the correct voltage. Once loaded, they will retain their values for at least ten years, unless they've been erased. Memory chips with capacitors are called EPROMs (Eraseable, Programmable Read Only Memory). These chips are often used in small production devices, such as interface boards. These can only be erased with great difficulty, and then, all information on the chip will be lost.

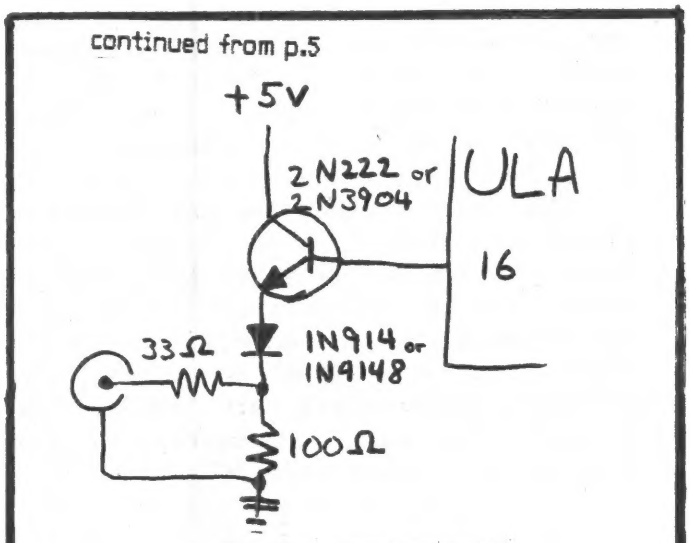
RAMs

Both devices above are somewhat like engraving in granite. It would be nice to have something more like a blackboard that would retain the information until it was specifically erased. The answer is the RAM. This is composed of transistor networks that will stay in an "on" state until specifically turned "off". This is the device that holds your BASIC programs and other information that can be changed at will. Unfortunately, these chips must constantly have a supply of +5V to keep the transistors happy. A very short interruption causes all the banks to return to zero.

This completes our tour of the levels of the Computer. Each level is adapted to dealing with both the level above, and the level below. In your relations with the computer, you can decide what level is most appropriate, from the most general canned program, to the most detailed hardware level. Fortunately, the computer is an amazingly forgiving device. Except for the hardware level, you can't hurt the machine with an incorrect command. Your program may crash, but the computer will patiently wait for you to try again.

Mark Fisher

continued from p.5



Monitor Amp Schematic

SUBROUTINE FOR MENU SELECTION
by
Mrs. Anne W. Andersen

This subroutine was written for the T/S 1000 in the BASIC language. It will also work on the T/S 1500, and probably on many other computers. It may be used to input any single-digit number (from zero through nine), perhaps after a menu has been printed.

This subroutine has some advantages compared to an INPUT instruction. With this subroutine, the user presses only one key per character (without pressing ENTER). Also, if the user does not wish to respond to the prompt, the BREAK key may be used to interrupt the program.

```
10 REM  INPUT SUBROUTINE.
11 IF INKEY$<>"" THEN GOTO 11
12 PRINT AT 21,1;"?      "
13 LET A$=INKEY$
14 IF A$="" THEN GOTO 13
15 PRINT AT 21,1;A$
16 IF A$<"0" OR A$>"9" THEN GOTO 11
17 LET I=VAL A$
19 RETURN
```

Line 11 makes the program wait, if necessary, until the system is ready to receive a new character from the keyboard. Line 12 prints a question mark near the lower left corner of the screen. It also erases the next five spaces, to finish deleting the previous value (even if it was a shifted command word). In line 12, I really use an inverted question mark -- but this printer has no inverse characters.

The user presses one key (without pressing ENTER). Line 13 gets this character and stores it in A$. Line 14 makes the program wait until the user has pressed a key. Line 15 prints the character, and line 16 tests it. Rejected characters are erased and ignored, instead of interrupting the program with error code "C".

Line 17 converts the character to a numeric value. Some programs may not need this line.

This routine was derived from a typewriter routine in the Timex User Manual. Many variations are possible. The location of the "cursor" (the question mark) could be moved by modifying the "21,1" in lines 12 and 15. Line 16 could be modified to agree with a different menu, provided each choice is represented by a single character. The character need not be numeric.

This routine is not designed for entering multiple-character values. It is usually best to use an INPUT instruction for such values. However, it would be possible to write a routine which would accept multiple-character values keyed without ENTER, provided the count of characters for that value is constant.

The first word on each line is a single-key instruction. In lines 11, 14, and 16, "GOTO" is a single-key command word, and "THEN" and "OR" are shifted single-key items. In line 11, the <> must be obtained by pressing shifted T. In line 12, I obtain an inverse question mark by pressing shifted 9 (GRAPHICS) to get the inverse C cursor, pressing the normal question mark (shifted C), and pressing the shifted 9 again to exit GRAPHICS mode. In lines 11, 13, 15, and 17, to obtain "INKEY$", "AT", and "VAL", press FUNCTION (shifted ENTER) to get the inverse F cursor first, and then press the single-key "word". Line 16 will occupy 2 lines on your screen; that's normal. (Everything in this paragraph applies to TIMEX BASIC only).

To use the subroutine, a GOSUB instruction is required. GOSUB 10 will work on my computer, but I use GOSUB 11 for efficiency.

If there are any questions about this subroutine, please contact me at Soft Answer, phone (301) 474-7922.

[Switching lines 11 and 12 may improve this. Ed.]

LESSIONS LEARNED ON THE USE OF JOYSTICK PORTS FOR I/O

1) Most joystick connector cables have only 6 wires, just those needed to allow the reading of up to 5 switches. If you want more than to just determine if a switch is closed, find a cable with all 9 conductors present.
(Philmore CE9 at Mark Elec. in Beltsville)

2) Not all 2068's have pin 9 at ground; mine does not, so I added a jumper from the dock connector ground to the right side port connector.

3) Data rates are rather low if the port is controlled by a BASIC program, probably less than 70 baud.

4) The port is set for input at power up. To switch to output, execute SOUND 7,64. The data is output by SOUND 14,data. To switch back to input mode, execute SOUND 7,0. To read data use STICK. (The BASIC commands IN and OUT can also be used; this involves more statements and is slower.) The value of the data byte of course depends on which pins you want to go high or low. E.G. pin 1 is bit 0; if only pin 1 is to be high then "data"=1.

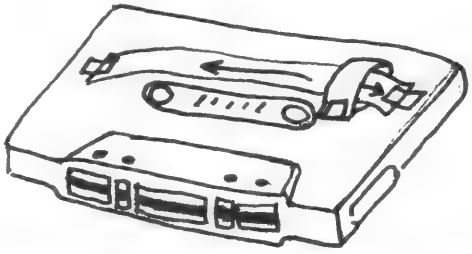
5) At power-up, all bits are set; e.g. a LED connected from pin 1 to ground will be lit. The STICK function will return 0. Now if you switch to the output mode and write a 0, all bits are reset, i.e. the LED goes out.

6) Any data written to the port is latched. If you later switch to the input mode and read some input data, it may reflect what you previously wrote. For example, write a 1, then read the port with STICK; the result is 14. This indicates that bit 0 is still set and the other bits are low, so the STICK functon thinks only pin 1 (bit 0) is not grounded.

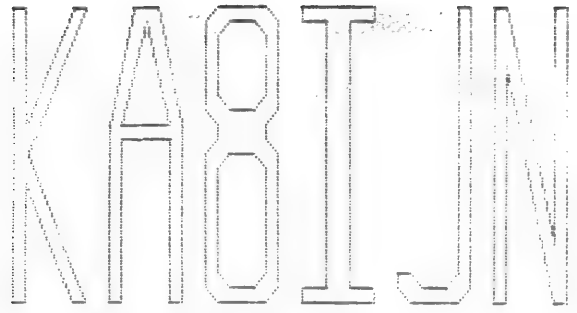
M. S. Morris

-CASSETTE REMINDERS-

I keep running total accounts SAVEd on my TS 1000. (Stock portfolio balances, fitness workouts, checkbook balances, etc) I hate to SAVE twice. If I save but once, every other day, I am only out one day if I goof. But, where is that last day? Pencil and paper I don't use. What's a computer (even TS 1000) for? I put two SAVES on one side of tape, and alternate between the two. To remember, I tape a strip of fimm paper to label part of the cassette from side to side. Then I loop another strip around that, fold flat and tape the end. Voila! Slide it from side to side to indicate where the last SAVE is (was?) Good for the absent-minded. Now all you have to do is remember to rewind to the beginning of the last SAVE so you start in the right place. I haven't found anything to help me with that. STEW VANCE



Washington Post 3/17/85



William R. Ware, Jr.
1237 Baltimore Annapolis Blvd.
Arnold, Maryland 21012

RADIO	DATE	UTC	RST	KHz
KA3000	1/31/85	1952	0	0

NICKNAME: BILL AGE: 42

TRANSMITTER: DRAKE 2-NT-XTAL

MODE: A-1

POWER OUTPUT: 0 WATTS

RECEIVER: HEATH HR-1680

ANTENNA: Mor-Gain 80-10M shunted dipole, 40 feet up.

[X] PSE QSL

qsl produced by

time x 2068

* this may be hard to read because the original was blue - bad news!

```
  5 REM ************************
 10 REM *       "Ham QSL"       *
 20 REM *     by Bill Ware      *
 25 REM *    1237 B-A Blvd.     *
 30 REM *  Arnold, Md. 21012    *
 35 REM ************************
 38 REM
 40 REM c$=Call sign
 50 REM m=month, d=date, y=year
 60 REM t=time(GMT),
        r=signal readability,
          strength and tone(599
 max.),  f=frequency in kilohert
z
 70 REM w=watts output
 200 INPUT "What is their call?:
";c$: INPUT "What is the number
of the month?:";m: INPUT "What i
s today's date?:";d
 210 INPUT "What is the year?:";
y: INPUT "What time is it?:";t:
INPUT "What is their RST?:";r: I
NPUT "What frequency are you usi
ng?:";f: INPUT "What is your pow
er output::";w: GO SUB 2000
1000 CLS : PRINT AT 0,0;"NICKNAM
E:  BILL";AT 0,22;"AGE:  42": PR
INT : PRINT "TRANSMITTER:  DRAKE
 2-NT-XTAL": PRINT : PRINT "
   MODE:  A-1": PRINT : PRINT "P
OWER OUTPUT:  ";w;" WATTS": PRIN
T : PRINT "RECEIVER:  HEATH HR-
1680": PRINT : PRINT "ANTENNA:
Mor-Gain 80-10M shunt-
ed dipole, 40 feet up."
1080 INPUT "QSL RECEIVED?(Y or N
):";a$
1100 PLOT 6,73: DRAW 10,0: DRAW
0,-10: DRAW -10,0: DRAW 0,10: GO
 SUB 2400
1200 PRINT AT 16,0;"qsl produced
";AT 18,5;"by";AT 20,1;"time x 20
68"
1600 COPY
1900 INPUT "DO YOU WISH TO MAKE
ANOTHER?:";C$: IF C$="Y" OR C$="
y" THEN GO TO 100
1945 LPRINT '';
1950 STOP
1999 REM [redacted]
2000 CLS : PLOT 10,75: DRAW 0,10
0: DRAW 5,0: DRAW 0,-40: DRAW 20
,40: DRAW 5,0: DRAW -24,-50: DRA
W 24,-50: DRAW -5,0: DRAW -20,40
: DRAW 0,-40: DRAW -5,0
2010 PLOT 50,75: DRAW 0,60: DRAW
 10,40: DRAW 10,0: DRAW 10,-40:
DRAW 0,-60: DRAW -5,0: DRAW 0,55
: DRAW -20,0: DRAW 0,-55: DRAW -
5,0: PLOT 55,135: DRAW 20,0: DRA
W -10,35: DRAW -10,-35
2020 PLOT 95,75: DRAW -5,5: DRAW
 0,45: DRAW 5,5: DRAW 0,5: DRAW
-5,5: DRAW 0,30: DRAW 5,5: DRAW
20,0: DRAW 5,-5: DRAW 0,-30: DRA
W -5,-5: DRAW 0,-5: DRAW 5,-5: D
RAW 0,-45: DRAW -5,-5: DRAW -20,
0
2030 PLOT 100,80: DRAW -5,5: DRA
W 0,35: DRAW 5,5: DRAW 10,0: DRA
W 5,-5: DRAW 0,-35: DRAW -5,-5:
DRAW -10,0: PLOT 100,140: DRAW -
5,5: DRAW 0,20: DRAW 5,5: DRAW 1
0,0: DRAW 5,-5: DRAW 0,-20: DRAW
 -5,-5: DRAW -10,0
2040 PLOT 130,75: DRAW 0,5: DRAW
 10,0: DRAW 0,90: DRAW -10,0: DR
AW 0,5: DRAW 30,0: DRAW 0,-5: DR
AW -10,0: DRAW 0,-90: DRAW 10,0:
 DRAW 0,-5: DRAW -30,0
2050 PLOT 175,75: DRAW -5,5: DRA
W 0,5: DRAW 5,0: DRAW 5,-5: DRAW
 10,0: DRAW 5,5: DRAW 0,90: DRAW
 5,0: DRAW 0,-95: DRAW -5,-5: DR
AW -20,0
2054 REM
2060 PLOT 210,75: DRAW 0,100: DR
AW 5,0: DRAW 20,-80: DRAW 0,80:
DRAW 5,0: DRAW 0,-100: DRAW -5,0
: DRAW -20,80: DRAW 0,-80: DRAW
-5,0
2070 PLOT 0,18: DRAW 255,0: DRAW
 -58,0: DRAW 0,-15: DRAW 58,0: D
RAW -90,0: DRAW 0,15: DRAW 0,-15
: DRAW -40,0: DRAW 0,15: DRAW 0,
-15: DRAW -72,0: DRAW 0,15: DRAW
 0,-15: DRAW -53,0
2100 PRINT AT 14,6;"William R. W
are, Jr.";AT 15,1;"1237 Baltimor
e Annapolis Blvd.";AT 16,5;"Arno
ld, Maryland 21012";AT 18,1;"RAD
IO";AT 18,9;"DATE";AT 18,16;"UTC
";AT 18,21;"RST";AT 18,27;"KHz";
AT 20,0;c$;AT 20,7;m;"/";d;"/";y
;AT 20,16;t;AT 20,21;r;AT 20,25;
f
2110 COPY : LPRINT
2120 RETURN
2400 IF a$="Y" OR a$="y" THEN PR
INT AT 13,1;"x";AT 13,3;"[redacted]
": RETURN
2410 PRINT AT 13,1;"X";AT 13,3;"
PSE QSL": RETURN
```

Thoughts We Have All Had and Never Expressed

March 5, 1985

Dear Mark Fisher:

My first problem is that I seem to be several laps behind in my knowledge of the TS problem. Since 2068, Spectrum and QL have come upon the scene, I have been unable to come to meetings because of conflicting family schedules on Saturdays (and besides, I'm way over here in Virginia).

While magazines seem to be able to label their differing articles on the 2068, Spectrum, QL, 1000, and 1500 separately and keep them separated, it appears that CATS Newsletter finds this impossible. What a pity. We retarded citizens have trouble knowing exactly which instructions we must follow. I use a TS 1000, and no other, so I have no need to read and understand the material on other machines that is not pertinent. I had thought I could devine between the TS 1000 and 1500 and others by the upper and lower case letters appearing in the programs, but now and then we find that someone has copied programs on other "word processors" (I think) and failed to honor the difference.

Could you publish a definitive article about the differences between the varous models and what can be transferred from one to another? That would help a lot. I can't contribute the article as I don't know, and it appears that those who do know aren't contributing. Needed is help.

I use my TS 1000 a lot. I have developed a library of financial calculation formulae which now are programs, and they help a lot. I also maintain day-by-day track of a stock portfolio through a program I have written, debugged and use although it still needs work and probably always shall. I doubt that my use and need will expand beyond the above. But, I am sympathetic to others' problems in deriving other uses from the little monster, and I am interested in others' interests and needs, so I cherish the CATS NEWSLETTER, and would like to come to meetings.

Well, best regards, and best of luck.

Yours,

Stew Vance

P.S. Examples;

CATS NEWSLETTER March 1985

page 11, "from T/S Users Group, ABILENE TEX. Paul Maserang.
What machine is he talking of? Obviously not 1000, but what? Also, that line "When saving a program to auto RUN by using LINK, if your first line is a BEEP" etc., is gobbledegook to a TS 1000 user without specialized knowledge.

page 12: "2068 Printing Calculator" line 2: it should work on the Timex 1000 if line 100 is changed." (Changed to what?) I challenge this to work with any unstated change on a Timex 1000. Example Line 66 or any other line which continues on without a new line after a colon. And, where did that

continued on p.14

continued from p.13

vertical arrow come from in line 250? line 940? Enough! Schrack, a volunteer, and not to be scorned, can't see beyond the end of his own nose. And, the newsletter editor is perhaps required to be an editor, not a reproducer. That's ppinter's job.

page 14: Ward Seguin's article on READ DATA Statements. What machine? I thought it was a continuation of the READ DATA in foregoing C.TS Newsletters, and for TS 1000, but that's t hat End(Upper and lower case) entry on line 70 doing? Writer or editor? Someone's at fault.

page 15: Ward Seguin's Loan Amortization. What machine? Sure a lot of unfamiliar items for a TS 1000 user.

page 14: "Your?" commentary on Read/DATA & Def FN: While this is addressed to ones who use TS 1000, they have no references to what you are talking about, as READ/DATA are not TS 1000 talk, nor is DEF FN.

MIRACLE TIME!

Fat Characters for your printer and your screen.

I'm happy to announce that that this thing WORKS!

The thing I'm referring to is the accompanying listing, which allows MSCRIPT to work with the FAT CHARACTER subroutine that was first developed by the Triangle Sinclair User's Group.

MSCRIPT uses the 64 column mode of the 2068. Normally, this means that each bit is only half as wide as normal, putting the character at the limit of resolution for the average monitor (and well beyond the limit for broadcast TV).

The following listing creates a new character file, with all vertical lines two bits wide. This can be used with either the 2040 printer and your normal programs, or MSCRIPT.

To use the routine to get enhanced printouts from your 2040 printer, type in lines 1 to 8, SAVE it,then MERGE with the program you'd like to enhance.

To use this with MSCRIPT, type in the full listing, SAVE it, then load MSCRIPT CODE. Now, for the first time, GOTO 1. The program will stall while trying to LOAD MSCRIPT again. BREAK, and GOTO 11; then press "b" and create a working hybrid word processor.

In use, the fatter letters are much easier to read. Only lower case "m" and "w" are dificult, and even they are better than they were. Go for it!

Great thanks are due to the person that worked this out. Unfortunately, the author was too modest to put his/her name in the first REM statement, and the cover letter was lost. Would the real author please stand up?

Mark Fisher

```
  1 REM MSCRIPT LOADER WITH FAT
 CHARACTERS
  3 CLEAR 64736
  5 DATA 017,000,253,213,001,00
0,003,042,054,092,036,126,167,03
1,182,018,035,019,013,032,246,01
6,244,225,037,034,054,092,201
  7 FOR I=64737 TO 64765: READ
N: POKE I,N: NEXT I
  8 RANDOMIZE USR 64737
  9 CLS : INK 6: PAPER 0: FLASH
 1: PRINT AT 10,5;"LOADING MSCRI
PT CODE": FLASH 0
 10 LOAD ""CODE
 11 CLS : PRINT AT 20,0;"PRESS
 'b' FOR BACK UP","ENTER TO START
 MSCRIPT"
 12 IF INKEY$="" THEN GO TO 12
 13 IF INKEY$="b" OR INKEY$="B"
 THEN GO TO 15
 14 GO TO 20
 15 CLS : PRINT AT 15,0;"MAKING
 BACK UP": SAVE "MSCRIPT" LINE 1
: PAUSE 100: SAVE "MSCRIPT"CODE
36864,7931: CLS : PRINT AT 15,0;
"REWIND AND PLAY TO VERIFY": VER
IFY "": VERIFY ""CODE : PRINT "V
ERIFY OK": PAUSE 200: GO TO 11
 20 DELETE 1,19
 21 CLEAR 64767: CLS : RANDOMIZ
E USR 36864
```

AVOIDING THE FUNCTION DISPATCHER

The 2068's function dispatcher is intended to give application programmers access to a number of ROM routines. These routines can be more easily accessed by a direct call. The addresses of all service routines can be found in Appendix A of the 2068 Technical Manual;e.g. service routine 26 has the name PARP, which according to the Appendix is at 03F3(hex).

Mike Morris

REPLY to STEW

You're right on target with a lot of what you say, Stew. I think your thoughts are shared by a number of other members. I applaud you for having the nerve to come out and name the troubles you're having.

1000 vs 2068 vs Spectrum, etc.

You are right that the CATS newsletter often fails to clearly describe the equipment used to develop the listings. It can be quite hard to tell them apart - and it dosen't need a word processor to make it so; just "CAPS LOCK" on the 2068. In my defense, I would say that you should consider whether you, in fact, have "no need to read and understand material on other machines that is not pertinent." You may find that your flexibility and control of your 1000 will grow by learning something of the quirks of those other machines. In fact, the Sinclair family are quite similar - there are just a few extra things to understand, coming from the 1000. Translating from other machines can be harder, especially when string handling (Sinclairese is much easier to use). But hard or easy, learning to adapt programs written from other machines can do wonders in expanding your horizons.

You listed some specific questions at the end of your letter; I've got some specific answers, and then I'll follow with some general principals.

1. page 11; BEEP isn't a working word for the 1000 - just look in the manual's index. There isn't any "special knowledge" that can evoke it. Seeing the word BEEP should be enough to drop the rest of the item.
2. page 12; The printing calculator piece was written for the 2068; translating it to the 1000 involves inserting extra lines for each colon, and changing the PAUSE statements from PAUSE 0 to PAUSE 4E4.Translation can be a valuable skill. The knowledge you'll need will be useful when dealing with programs from a variety of sources.
3. & 5., page 14; Ward's article covered general qualities of the READ/DATA command. While the 1000 doesn't have those keywords, they can be simulated, as discussed in my following article. Ward's use of trailer data works with my routine as well. My READ/DATA article was an abridged recap of an earlier article - I can arrange for a copy of the original, if you like. If you had punched it in, I think that you would have understood what was meant.
4. page 15; that's right, it's for the 2068. But 99% of it can be directly typed into the 1000, and the rest is easily adapted. See below.

Signals that it ain't for the 1000 (and tips for translation)

If you see any of the following, you can be sure that the listing wasn't from a 1000. The suggested translations aren't the last word - see item 6.

1.Multi-statement lines. Cure by writing multiple lines. If there is an IF in the line, all statements after it will be ignored if it is false. Cure by using a succession of IF lines, or by jumping to a subroutine.

2. PAUSE 0. Use PAUSE 4E4. PLOT is another case of identical commands, with different parameters: the 2068's screen is 254x175, while the 1000's is 44x64.

3.Lower case. Not infallible - see line 9900 of this month's "AMORTIZE". CAPS LOCK on the 2068 can fool you also.

4. Apostrophes. The ' is used in the 2068 to drop one line. In last month's PRINTING CALCULATOR, multiple PRINTs were used. Use PRINT on a seperate line, or ":".

5. Funny operators: ↑, READ/DATA, DEF FN, VAL$, STICK, BEEP, DRAW, etc. ↑is **. Sometimes the others can be imitated by subroutines. READ/DATA can be used on the 1000 as I discussed last month. VAL$ can't, but it is rarely used. STICK can be replaced by INKEY$. BEEP, SOUND, etc. can be ignored. CIRCLE and DRAW can be replaced by subroutines (I've got a good machine code DRAW, if anybody wants it).

6. There are probably more, but I can't think of them right now. If you readers have any further tricks of translation, send them in - they'll be printed.

AMORTIZE for the 1000

You may find the following program to be of help as an example of translation from the 2068 to the 1000. It is a rework of Ward's Loan Amortization program in last month's issue. Since the 2068 transcends SCROLLs, I had to put them back in the program. The other additions were:

1. Schedule starts on any given month. New variable ST. (stolen from Albert Strauss).

2. Yearly breakouts of interest and principal. New variables YI, YP, and YT.

3. Echo to printer (turn printer off to disable, or delete lines). LPRINT lines are set up for an 80 column printer - if you've got a 2040, make

```
 10 REM *** AMORTIZATION ***
            EQUAL PAYMENTS
 20 REM BY W SEGUIN, 1985 - TRA
NSLATED TO T/S1000 BY M FISHER
 30 PRINT "AMORTIZATION TABLE"
 40 LPRINT "AMORTIZATION TABLE"
 50 PRINT ,,"AMOUNT BORROWED ="
;
 60 LPRINT "AMOUNT BORROWED =";
 70 INPUT P
 80 PRINT P
 90 LPRINT P
100 PRINT ,,"NO. OF YEARS TO PA
Y =";
110 LPRINT "NO. OF YEARS TO PAY
 =";
120 INPUT Y
130 PRINT Y
140 LPRINT Y
150 PRINT ,,"STARTING AT MONTH
NO. (1 TO 12) ";
160 LPRINT "STARTING AT MONTH N
O. (1 TO 12) ";
170 INPUT ST
180 PRINT ST
190 LPRINT ST
200 REM COMPUTE NO. OF MONTHS
210 LET M=12*Y
220 PRINT ,,"INTREST/YR (6 1/2
PCT=6.5)=";
230 LPRINT "INTREST/YR (6 1/2 P
CT=6.5)=";
240 INPUT I
250 PRINT I
260 LPRINT I
270 REM MONTHLY INT
280 LET R=I/(100*12)
290 REM EQUAL MONTHLY PAYMENTS
300 LET M1=(1+R)**M
310 LET E=(P*R*M1)/((1+R)**M-1)
320 LET E=(INT (E*100+.5))/100
330 REM CALC, THEN PRINT
340 LET TI=0
350 LET TP=TI
360 LET SP=TI
370 LET YI=TI
380 LET YP=TI
390 LET YT=TI
400 PRINT ,,,,
410 PRINT "MONTHLY PAYMENT =";E
420 LPRINT "MONTHLY PAYMENT =";
E
430 SCROLL
440 PRINT "MO  PRINCIPAL INTERE
S  PRINCIPL"
450 LPRINT "PAY MO  PRINCIPAL I
NTEREST  PRINCIPL"
460 SCROLL
470 PRINT "NO    OWED      PAYMN
T    PAYMT"
480 LPRINT "NO  NO    OWED
PAYMNT    PAYMT"
490 REM THEN PRINT DASHES
500 SCROLL
510 PRINT "----------------------
------------"
520 LPRINT "----------------------
----------------"
530 FOR J=1 TO M
540 LET I1=P*R
550 LET P1=E-I1
560 IF J=M THEN LET P1=P
570 IF J=M THEN LET I1=E-P1
580 LET I1=(INT (I1*100+.5))/10
0
590 LET P1=(INT (P1*100+.5))/10
0
600 SCROLL
610 LPRINT J;TAB 4;ST;TAB 9;P;T
AB 20;I1;TAB 29;P1
620 PRINT ST;TAB 5;P;TAB 16;I1;
TAB 25;P1
630 REM COMPUTING TOTALS
640 LET TI=TI+I1
650 LET YI=YI+I1
660 LET TP=TP+P1+I1
670 LET YT=YT+P1+I1
680 LET SP=SP+P1
690 LET YP=YP+P1
700 LET P=(INT ((P-P1)*100+.5))
/100
710 IF ST=12 THEN GOSUB 910
720 REM YEAR TOTALS
730 LET ST=ST+1
740 IF ST=13 THEN LET ST=1
750 NEXT J
760 GOSUB 910
770 REM COMPUTING SUMMARY TOTAL
S
780 LET TI=(INT (TI*100+.5))/10
0
790 LET SP=(INT (SP*100+.5))/10
0
800 LET TP=(INT (TP*100+.5))/10
0
810 SCROLL
820 PRINT "TOTAL INTEREST =";TI
830 LPRINT "TOTAL INTEREST =";T
I
840 SCROLL
850 PRINT "TOTAL PRINCIPAL=";SP
860 LPRINT "TOTAL PRINCIPAL =";
```

```
S
 870 SCROLL
 880 PRINT "TOTAL PAYMENTS =";TP
 890 LPRINT "TOTAL PAYMENTS =";T
P
 900 STOP
 910 REM **** YEARLY SUBS ****
 920 SCROLL
 930 PRINT "YEAR NO. ";INT (J/12
)+1;" TOTALS:"
 940 LPRINT "YEAR NO. ";INT (J/1
2)+1;" TOTALS:"
 950 SCROLL
 960 PRINT "    TOTAL    INTEREST
  PRINCIPL"
 970 LPRINT "    TOTAL    INTERES
T  PRINCIPL"
 980 SCROLL
 990 PRINT "    PAID      PAYMNT
    PAYMT"
1000 LPRINT "    PAID      PAYMNT
     PAYMT"
1010 SCROLL
1020 LPRINT TAB 3;YT;TAB 14;YI;T
AB 22;YP
1030 PRINT TAB 3;YT;TAB 14;YI;TA
B 22;YP
1040 LET YI=0
1050 LET YP=YI
1060 LET YT=YI
1070 SCROLL
1080 PRINT "-------------------
------------"
1090 SCROLL
1100 PRINT "MO   BALANCE  INT. P
Y    PRINC."
1110 LPRINT "NO  MO   BALANCE  I
NT. PY    PRINC."
1120 SCROLL
1130 PRINT "-------------------
------------"
1140 LPRINT "-------------------
-------------"
1150 RETURN
9000 LET X=16509
9010 LET N=10
9020 IF PEEK X*256+PEEK (X+1)>=9
000 THEN STOP
9030 POKE X,INT (N/256)
9040 POKE X+1,N-(PEEK X*256)
9050 LET X=X+PEEK (X+2)+PEEK (X+
3)+4
9060 LET N=N+10
9070 GOTO 9020
9900 SAVE "AMORTIZe"
9910 RUN
```

each LPRINT identical to the PRINT line near it.

To simplify the typing, DEF FN could be added by adding a line:

```
5 LET F$="INT (X*100+.5)/100"
```

and replacing each incidence of the string with two lines: ex:

```
575 LET X=I1
580 LET I1=VAL F$
```

Lines 9000 to 9070 are a quick re-numbering utility that I put in to clean up my print out [see Hacking in BASIC, this issue].

Mark Fisher

DOUBLE PRECISION

This code will do "double precision" -that is it will take two 8-digit numbers and give a 16-digit number. It accomplishes this by breaking each 8-digit number into two 4-digit numbers and then multiplying them. The products are then suitably re-combined.

```
  5 REM   DOUBLE PRECISION
 10 PRINT "Give first number (<
=8 digits)": INPUT a: PRINT a: P
RINT
 20 PRINT "Give second number (
<=8 digits)": INPUT b: PRINT b:
PRINT
 22 LET ah=INT (a/10↑4): LET al
=INT (a-10↑4*ah+.1)
 24 LET bh=INT (b/10↑4): LET bl
=INT (b-10↑4*bh+.1)
 30 LET chh=ah*bh
 40 LET chl1=ah*bl
 42 LET chl2=al*bh
 50 LET cll=al*bl
 60 LET chhh=INT (chh/10↑4): LE
T chhl=INT (chh-10↑4*chhh+.1)
 70 LET chlh1=INT (chl1/10↑4):
LET chll1=INT (chl1-10↑4*chlh1+.
1)
 72 LET chlh2=INT (chl2/10↑4):
LET chll2=INT (chl2-10↑4*chlh2+.
1)
 80 LET cllh=INT (cll/10↑4): LE
T clll=INT (cll-10↑4*cllh+.1)
 90 LET c1=10↑4*chhh+chhl+chlh1
+chlh2
 100 LET c2=10↑4*(chll1+chll2+cl
lh)+clll
 102 IF c2<10↑8 THEN GO TO 110
 104 LET c2=10↑4*((chll1-5000)+(
chll2-5000)+cllh)+clll
 106 LET c1=c1+1.
 110 LET z$="0000000"
 112 FOR x=1 TO 7
 114 IF (c2<10↑x) THEN GO TO 122
 116 NEXT x
 118 PRINT c1;c2
 120 STOP
 122 LET w$=z$(x TO )
 124 PRINT c1;w$;c2
```


ROALD A. Schrack

HACKING in BASIC

RENUMBERING INCLUDING GOTO and GOSUB

There have been several examples of renumbering routines that have run in the CATS newsletter over the years. They have all shared a common failing - they couldn't alter the GOTOs and GOSUBs. This is a problem in BASIC, because all program jumps (GOTO & GOSUB) are absolute jumps. The program jumps to a specific line number, not a specific number of lines foreward. When a program is renumbered, a given number may refer to a very different line in the program.

In thinking about the problem, it seemed that the only way to change the GOTOs was via machine code. Programs such as this are available, called toolkits. But even the best of the toolkit programs can't handle renumbering computed GOSUBS, such as GOSUB 500*X+1000.

Then, a couple of weeks ago, I realized that there is a way to change ALL GOTOs and GOSUBs from absolute jumps, to relative jumps. And it's all in BASIC, on either the 1000 or the 2068! With relative jumps, a block of instructions can be changed from lines 100-300 to 400-600, while a GOTO that jumps ahead 3 lines (or 30) continues to work.

For the 1000: start your program with

```
1 LET F$="PEEK 16391+256* PEEK 16392"
```

After running this line, PRINT VAL F$ will return the current line number! This takes advantage of the system variable PPC, that holds the line number being executed. An absolute jump to two lines ahead might be written

```
20 GOTO 40
```

(assuming you are using intervals of ten for each line). To change this to a relative jump to two lines ahead, substitute:

```
20 GOTO VAL F$+20
```

As written, it will GOTO line 40. If it is renumbered to line 80, it will GOTO line 100.

Computed GOTOs are also possible, as in GOSUB VAL F$*500+1000. The only time this will fail, is if you change the numbering interval, or add a line of code within the numbers skipped by the GOTO. Obviously, this is more likely for large jumps than small - but it's the small jumps that often just can't be avoided.

For the 2068: the 2068 has a real DEF FN command, and this is a perfect place to use it:

```
1 DEF FN f()=PEEK 23621+256*PEEK 23622
```

followed by

```
20 GOTO FN f()+20
```

Try this out for yourself - a renumbering program is hidden in lines 9000 to 9070 of the AMORTIZER program in this issue.

Mark Fisher

UNCLASSIFIED

FOR SALE: T/S 2068 & Westridge MODEM. Call Richard,Roland, or Constance Roseen 703-527-8467 or 703-533-3590

FOR SALE: 2 T/S 2068's, 2 cassette recorders, 2 2040 printers, + software. No reasonable offer refused. Mark [no relation]; 548-7615 (H) 344-5819 (W)

FREE: Excellent catalog. Write Sunset Electronics, 2254 Taraval St., San Fransisco CA 94116

FOR SALE: Typewriters - IBM Selectric I $250 & IBM Model C $75.00. No reasonable offer refused. 301-559-4281.

MRI books offers a catalog of 1500 computer books (including Mastering Machine Code), free. MRI Books, Alexandria VA, 549-5537

For parts for all those hardware projects, try: Revacto or Heathkit in Rockville, Electronics Plus in College Park, or Mark [no relation] Electronics in Beltsville.

Thoughts on MSCRIPT

I've used MSCRIPT to write several of the articles in this issue. There has been a lot of learning that I had to do. In general, I agree with the review we ran last month. There have been some surprises.

1. It is more easily used that I feared. I've diddled with WORDSTAR, and have yet to feel comfortable with it. MSCRIPT is all right.
2. As it comes, it is hard to read, even on a monitor. Fat Characters totally fixes this.
3. Its tape SAVEs and LOADs are its biggest weakness. It dosen't use the standard T/S routines, and will not load "". Most importantly, it cannot VERIFY. I've already lost several files due to glitches of some sort, on good tape. (Learning goes on - I just realized that I could use the Append command to verify that a file could be reloaded, without destroying the original.)
4. Its operation is smooth. This is a professionaly done program, and it shows. It is tolerant of variations in commands: LM=9, lm=9, and lm9 all do the same thing. Its HI command (hanging indent) is a treasure. I'm using it now.
5. The manual is easy to read and comprehensive. It does not include a command summary for its numerous commands.

Rating: a slightly tarnished *****.

Capitol Area Timex/Sinclair Users' Group
P.O.Box 725
Bladensburg, MD 20710

Name _____

Address _____

_____ ZIP _____

Phone Home _____ Office _____

Memberships - $15.00 (family/individual);make checks payable to C.A.T.S.

If family membership, please list family members participating:

_____

Occupation _____

Ham Radio call sign _____

Equipment

ZX 80 _____ RAM size _____

MA 80 _____ full keyboard _____

ZX 81 _____ Printer _____

TS 1000 _____ type _____

TS 2000 _____ other interface _____

Special interest use for computer: ie, games, ham radio interface, business, other, etc. _____

_____

Languages: Basic _____ Other _____

Machine _____ _____

No. of years computer experience _____

What committees would you like to serve on? _____

_____

Comments: _____

_____

- -

Do not write below:

Dt. Pd. _____ Amt. _____ Membership No. _____

Ca. _____ Ck. _____

The mailing address of the Capitol Area Timex/Sinclair User's Group is:

Capitol Area Timex/Sinclair User's group
P.O. Box 725
Bladensburg, MD 20710

CATS is a non-profit special interest organization dedicated to serving the interests of those who own, use, or are interested in learning more about the Timex/Sinclair family of personal computers.

The official contact person for CATS is JULES GESANG:

301*922-0767

Meetings are held on the second Saturday of each month at 2 P.M. in the large meeting room of the New Carrollton Branch Public Library.

Ham Radio Network Information
QZX Net...Wednesdays, 9p.m. local time; 14.345 MHz NV4F NCS
Eastern Regional Sinclair Net...Sundays, 1600 Z; 7.245 MHz KQ2F NCS

CATS Newsletter
P.O. Box 725
Bladensburg MD 20710

BULK RATE
U. S. POSTAGE
PAID
Permit No. 40
RANDALLSTOWN, MD.

The next meeting of CATS will be held on:
********* SATURDAY, April 13, 1985 @ 2:00 PM

New Carrolton Public Library
7414 Riverdale Road, New Carrolton, MD

IF YOU ARE NOT A MEMBER OF CATS, THIS IS THE ONLY ISSUE YOU WILL RECIEVE

Dues = $15.00 per year, per family.

DATED MATERIAL